Kvm_open

Be very careful with privilege mode of calling process

Sean Barnum, Cigital, Inc. [vita[1]]

2007-03-26

Part "Original Cigital Coding Rule in XML"

Mime-type: text/xml, size: 7055 bytes

Attack Category	• Privilege Exploitation
Vulnerability Category	• Privilege escalation problem • Indeterminate File/Path • TOCTOU - Time of Check, Time of Use
Software Context	• Memory Management
Location	• kvm.h
Description	When kvm_open() or kvm_openfiles() are used in a privileged program, care must be taken to ensure that no inappropriate access is granted to a user. The functions kvm_open() and kvm_openfiles() return a descriptor used to access kernel virtual memory via the kvm library routines. Both active kernels and crash dumps are accessible through this interface. The information provided by the kvm library routines is potentially of use to hackers trying to craft an attack against the examined code. It is therefore important that, if the process is running in a privileged mode (e.g., suid-root), care be taken to ensure that users are not given access to kernels to which they should not have access. The opened kernel or crash dump is specified by its file path. This means that in some cases, these functions may be vulnerable to path related attacks.
APIs	**Function Name** / **Comments** kvm_open kvm_openfiles
Method of Attack	If an attacker can run an suid-root program that examines kernel virtual memory and gain access to this information, then the attacker may gain information useful in exploiting other vulnerabilities. The key issue with respect to TOCTOU vulnerabilities is that programs make assumptions

1. http://buildsecurityin.us-cert.gov/bsi-rules/35-BSI.html (Barnum, Sean)

about atomicity of actions. It is assumed that checking the state or identity of a targeted resource followed by an action on that resource is all one action. In reality, there is a period of time between the check and the use that allows either an attacker to intentionally or another interleaved process or thread to unintentionally change the state of the targeted resource and yield unexpected and undesired results.

Even if the program invoking kvm_open() first does a check to ensure that permissions are appropriate, the attacker may be able to exploit a race condition to substitute a different kernel for examination before kvm_open() is called.

Exception Criteria

Solutions

Solution Applicability	Solution Description	Solution Efficacy
Whenever kvm_open() or kvm_openfiles() is used.	If program is running in a privileged mode such as suid-root, before calling kvm_open() the effective user and group IDs should be set to reflect the actual user, to ensure that no inappropriate access is granted.	Typically effective.
Generally applicable.	The most basic advice for TOCTOU vulnerabilities is to not perform a check before the use. This does not resolve the underlying issue of the execution of a function on a resource whose state and identity cannot be assured, but it does help to limit the	Does not resolve the underlying vulnerability but limits the false sense of security given by the check.

		false sense of security given by the check.	
	Generally applicable.	Limit the interleaving of operations on files from multiple processes.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
	Generally applicable.	Limit the spread of time (cycles) between the check and use of a resource.	Does not eliminate the underlying vulnerability but can help make it more difficult to exploit.
	Generally applicable.	Recheck the resource after the use call to verify that the action was taken appropriately.	Effective in some cases.

Signature Details	kvm_t * kvm_open(const char *execfile, const char *corefile, char *swapfile, int flags, const char *errstr); kvm_t * kvm_openfiles(const char *execfile, const char *corefile, char *swapfile, int flags, char *errbuf);

Examples of Incorrect Code	`// running as suid-root`
	`kvm_t *handle = kvm_open(execfile, corefile, swapfile, flags, errstr);`
	`// then expose results of kernel virtual memory examination`

Examples of Corrected Code	`// running as suid-root`
	`// change effective user and group IDs to reduce privileges`
	`kvm_t *handle = kvm_open(execfile, corefile, swapfile, flags, errstr);`
	`// expose results of kernel virtual memory examination only as strictly necessary`

Source References	• ITS4 Source Code Vulnerability Scanning Tool[2] • http://www.phrack.org/show.php?p=60&a=6 • http://seclab.cs.ucdavis.edu/projects/vulnerabilities/scriv/ucd-ecs-95-09.pdf[4]
Recommended Resource	• man page for kvm_open(), kvm_openfiles()[5]
Discriminant Set	**Operating System**: • UNIX **Languages**: • C • C++

Cigital, Inc. Copyright

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

1. mailto:copyright@cigital.com